Amendments to the Specification

Please delete the paragraph starting at page 13, line 1.

Please replace the paragraph starting at page 14, line 9, with the following amended paragraph:

As is known in the art, Java is an object-oriented programming language developed by Sun Microsystems of Mountain View, California. ~~More information about Java can be found at the URL "java.sun.com."~~ Java is based on object-oriented programming techniques. As is known in the art, object-oriented programming is used to design computer software including object-oriented objects that are easy to create, cost effective to modify, and reusable. Object-oriented objects include "object data" and "object services." Object services are provided through "object methods" (also called "object operations" or "object functions"). Object methods typically operate on private data such as "instance data" or "object state data" that an object owns. A collection of objects is called an "object class" which is sometimes called an "object type." An object class acts as a template that describes the behavior of sets of objects. An object's implementation is typically encapsulated, and is hidden from public view. Object private instance data can only be accessed by object methods of an object class. Object public instance data is accessed through a public "object interface."

Please replace the paragraph starting at page 15, line 10, with the following amended paragraph:

The Java 2 Platform Micro Edition ("J2ME") is used to create applications for fixed and mobile wireless devices. J2ME is a subset of the Java 2 Platform Standard Edition ("J2SE"). ~~More information about J2ME can be found at the URL "java.sun.com/j2me."~~

Please replace the paragraph starting at page 27, line 21, with the following amended paragraph:

In one embodiment of the present invention, the object-oriented application program interface is a J2ME API 88. ~~called "com.sprintpcs.util."~~ However, the present invention is not limited to this exemplary J2ME API and other J2ME APIs with other monikers can also be used.

Please replace the paragraph starting at page 28, line 3, with the following amended paragraph:

FIG. 7 illustrates the exemplary ~~"com.sprintpcs.util"~~ API 88. This API 88 includes a System object class 90 and a Muglet object class 92. However, the present invention is not limited to this embodiment and other embodiments with more or fewer object classes can also be used. In addition, the API 88 and the object-classes included therein can use any monikers and are not limited to the monikers described.

Please replace the paragraph starting at page 29, line 1, with the following amended paragraph:

An *opaque* URI is an absolute URI whose scheme-specific part does not begin with a slash character ("/"). Opaque URIs are not subject to further parsing. Some examples of opaque URIs are illustrated in Table 5.

~~mailto:user@sprint.com~~	~~// mail~~
news:comp.lang.j2me	// news

Table. 5

Please replace the paragraph starting at page 29, line 6, with the following amended paragraph:

A *hierarchical* URI is either an absolute URI whose scheme-specific part begins with a slash character ("/") or a relative URI, that is, a URI that does not specify a scheme. A hierarchical URI is subject to further parsing. Table 6 illustrates a few examples of hierarchical URIs.

~~http://sun.java.com/j2me~~
docs/guide/collections/j2me
../../../sprint/demo/j2me
file:/../~calendar

Table 6.

Please replace the paragraph starting at page 30, line 17, with the following amended paragraph:

The object-method appendReferringURI( ) sets a string that is passed to the JAM 58 and appended to the URI identifying a MIDlet, when the MIDlet invokes another MIDlet or another application through the setExitURI( ) object method. The object method setExitURI( ) sets a URI that ~~what~~ is passed to the JAM 58 and invoked according to the rules of URI scheme and Internet media type processing.

Please add the following NEW paragraph after Table 8 on page 34 and prior to the section heading "Setting Output Data from MIDlets" on page 34:

A method of the invention includes passing input data to a first Java MIDlet in a first MIDlet suite on a mobile information device. Passing the input data to the first Java MIDlet includes (i) receiving from the first Java MIDlet a request for the input data via at least one of getMediaType(), getContentType(), getMuglet(), getReferringURI(), and getURI object-oriented methods, and (ii) responsively passing the input data to the first Java MIDlet. Another method of the invention includes passing the input data to a second Java MIDlet. Passing the input data to the second Java MIDlet includes: (i) receiving from the second Java MIDlet a request for the input data via at least one of getMediaType(), getContentType(), getMuglet(), getReferringURI(), and getURI() object-oriented methods, and (ii) responsively passing the input data to the second Java MIDlet.

Please replace the paragraph starting at page 36, line 15, with the following amended paragraph:

Method 100 is illustrated with an exemplary embodiment. However, the present invention is not limited to this embodiment and other embodiments can also be used to practice the invention. In such an exemplary embodiment at Step 102, ~~100,~~ a J2ME MIDlet is invoked on the mobile information device 12 from the JAM 58. The MIDlet has the four object-oriented methods from the Muglet 92 object class available for using input data including a URI scheme or an Internet media type (e.g., MIME type) created by other MIDlets. At Step 104, input data including a URI scheme or an Internet media type created by another MIDlet or a non-MIDlet application is accepted from the JAM 58 on the MIDlet using one or more of the object-oriented methods from Muglet 92 object class.

Please replace the paragraph starting at page 37, line 21, with the following amended paragraph:

A MIDlet intended to process one or more type of URI scheme or Internet media type can be invoked as a MIDlet handler via the Muglet object class 92. A Muglet is constructed as a standard MIDlet, and may make use of the full range of features available to MIDlets. Additionally, such a Muglet enumerates which URI schemes and Internet media types are handled by including certain properties in a corresponding ~~corrsponding~~ MIDlet Suite. When installed into a mobile information device 12, MIDlet Suite properties are used to configure the Muglet(s) to handle the specified URI schemes ~~chemes~~ and Internet media types for other MIDlets and non-MIDlet applications on the mobile information device 12. Such a model may be similar to how common desktop web-browsers use 'plug-in' modules and external applications.

Please replace the paragraph starting at page 43, line 6, with the following amended paragraph:

The entire URI scheme will be passed to the Muglet by the JAM 58 via the Muglet.getURI() object-oriented method. For example, a ~~the~~ URI string[[:]] including [[<a href=]]"midlet:com.sprintpcs.apps.calendar?date="20011003"[[>]] would launch a calendar MIDlet from "com.sprintpcs.apps" with the date set to October 3, 2001, from a Muglet.

Please replace the paragraph starting at page 43, line 11, with the following amended paragraph:

The JAM 58 also supports the ability to "launch" stored non-java content by handling URI string with a scheme of ("ams:") and a scheme-specific part that is a content-ID of the target content. For example, a ~~the~~ URI string including [[<a href=]]"ams:sprintpcs.com.example_content"[[>]] in a browser page would cause an ams: URI to be passed to the AMS, which would launch an appropriate handler for the stored content with a content-ID associated with "example_content."

Please replace the paragraph starting at page 47, line 7, with the following amended paragraph:

As shown in FIG. 11, the application management system 150 launches MIDlet C 158 to handle the output data set by MIDlet A 152. At dataflow 160, MIDlet C 158 ~~156~~ uses the getURI() object-oriented method to obtain the output data set by MIDlet A 152. In response, the application management system 150 returns to MIDlet C 158 the "A" output data, shown generally by dataflow 162.